

(19)日本国特許庁 (JP)

## (12) 公表特許公報 (A)

(11)特許出願公表番号

特表2002-521745

(P2002-521745A)

(43)公表日 平成14年7月16日(2002.7.16)

(51)Int.Cl.<sup>7</sup>

G 0 6 F 9/54

13/00

識別記号

5 3 0

FI

G 0 6 F 13/00

9/06

テーマコード(参考)

5 3 0 A 5 B 0 7 6

6 4 0 C

審査請求 未請求 予備審査請求 有 (全 42 頁)

(21)出願番号 特願2000-561545(P2000-561545)  
 (86)(22)出願日 平成11年7月15日(1999.7.15)  
 (85)翻訳文提出日 平成13年1月22日(2001.1.22)  
 (86)国際出願番号 PCT/US99/16055  
 (87)国際公開番号 WO00/05637  
 (87)国際公開日 平成12年2月3日(2000.2.3)  
 (31)優先権主張番号 09/120, 575  
 (32)優先日 平成10年7月22日(1998.7.22)  
 (33)優先権主張国 米国(US)

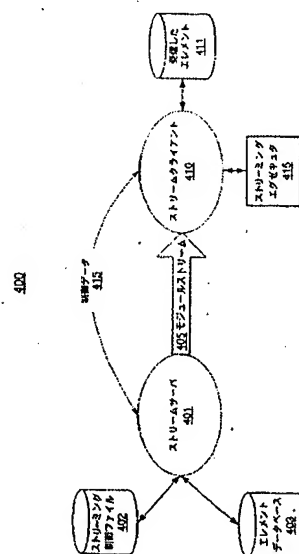
(71)出願人 アプストリーム インコーポレーテッド  
 アメリカ合衆国, カリフォルニア州  
 94303-3210, パロ アルト, イースト  
 ベイショア ロード 2595  
 (71)出願人 ユリ ラズ  
 アメリカ合衆国 07410 ニュージャージ  
 ー州 フェアローン ヒルサイド テラス  
 36-02  
 (71)出願人 ヤフーダ フォルク  
 イスラエル テルーアビブ レンブラント  
 ストリート 10  
 (74)代理人 弁理士 谷 義一 (外2名)

最終頁に続く

(54)【発明の名称】 モジュールのストリーミング

## (57)【要約】

第1コンピュータと第2コンピュータとの間でモジュールを伝送するためのコンピュータ実装方法が開示される。第1コンピュータでは、モジュールセットは利用可能なモジュールのコレクションからモジュールのシーケンスを選択することにより形成される。選択されたモジュールのそれぞれは、第2コンピュータで動作しているアプリケーションと関連付けられている。選択されたモジュールは次に、第1コンピュータから第2コンピュータに透過的にストリーミングできる。モジュールの選択は、予め定められた選択基準に従って行われ、第2コンピュータの実行環境とは独立している。第2コンピュータで、受信されたモジュールが、実行中のアプリケーションに統合され得る。第1コンピュータと第2コンピュータとの間でモジュールを伝送するためのシステムもまた提示される。開示のシステムは、第1コンピュータおよび第2コンピュータを含む。第1コンピュータは、アプリケーションを実行するための手段、アプリケーションが動作している間に、アプリケーションと関連付けられたモジュールのシーケンスを受信するための手段、お



**【特許請求の範囲】**

**【請求項1】** アプリケーションと関連付けられたモジュールのコレクションをストアする第1コンピュータから、前記アプリケーションを実行するための実行環境を提供する第2コンピュータにモジュールを伝送するコンピュータ実装方法であって、

前記実行環境とは独立している予め定められた基準に従って、前記コレクションからモジュールのシーケンスを選択することによってモジュールセットを形成し、

前記第1コンピュータから前記第2コンピュータにモジュールセットを透過的にストリーミングすることを特徴とする方法。

**【請求項2】** 透過的にストリーミングする際に、前記第2コンピュータでアプリケーションを実行している間に、前記第1コンピュータから前記第2コンピュータにシーケンスでモジュールを送信することを特徴とする請求項1に記載の方法。

**【請求項3】** 前記コレクション内の少なくとも1つのモジュールは、非実行データを備えることを特徴とする請求項1に記載の方法。

**【請求項4】** 前記コレクション内の少なくとも1つのモジュールは、プログラムコードを備えることを特徴とする請求項1に記載の方法。

**【請求項5】** 前記予め定められた基準は、ストリーミング制御データベースを備えることを特徴とする請求項1に記載の方法。

**【請求項6】** 前記ストリーミング制御データベースは遷移レコードを備え

前記遷移レコードは、前記コレクション内の選択されたモジュールの遷移に加重された値を関連付け、

シーケンスを選択する際に、遷移レコード値に基づいてモジュールの順序付けられたセットを選択することを特徴とする請求項5に記載の方法。

**【請求項7】** 遷移レコード値に基づいて選択する際に、パス決定アルゴリズムを使用して遷移レコード情報を処理することを特徴とする請求項6に記載の方法。

【請求項 8】 前記ストリーミング制御データベースは、複数のリスト記録を備え、

前記複数のリスト記録のそれぞれは、前記コレクション内のモジュールのシーケンスを識別し、

シーケンスを選択する際に、前記複数のリスト記録のうちの 1 つを選択することを特徴とする請求項 5 に記載の方法。

【請求項 9】 モジュールのシーケンスを選択する際に、前記第 2 コンピュータで選択し、透過的にストリーミングする際に、

前記コレクション内の第 1 モジュールを識別する選択データを第 1 コンピュータで受信し、

前記第 1 モジュールを第 2 コンピュータに伝送し、

前記コレクション内の第 2 モジュールを識別する第 2 選択データを前記第 1 コンピュータで受信し、

前記第 2 モジュールを前記第 2 コンピュータに伝送することを特徴とする請求項 1 に記載の方法。

【請求項 10】 さらに、前記モジュールセットの透過的なストリーミングに割込みを行い、

前記予め定められた基準に従ってモジュールの第 2 シーケンスを選択することによって第 2 モジュールを形成し、

前記第 1 コンピュータから前記第 2 コンピュータに前記第 2 モジュールセットを透過的にストリーミングすることを特徴とする請求項 1 に記載の方法。

【請求項 11】 さらに、前記第 2 コンピュータから前記第 1 コンピュータにステータスデータを伝送し、

前記ステータスデータに基づいて前記第 2 モジュールセットを決定することを特徴とする請求項 10 に記載の方法。

【請求項 12】 前記ステータスデータは、ユーザ入力を示すデータを備えることを特徴とする請求項 10 に記載の方法。

【請求項 13】 前記割込みの際に、

前記第 2 コンピュータから前記第 1 コンピュータにモジュール要求を送信し、

前記第1コンピュータから前記第2コンピュータに要求されたモジュールを送送することを特徴とする請求項10に記載の方法。

【請求項14】 前記モジュール要求は、前記モジュール内のコードを実行しようとする試みに応答して送信されることを特徴とする請求項13に記載の方法。

【請求項15】 プログラムはJavaアプレットであり、  
モジュール要求を送信し、要求されたモジュールを送送する際に、前記第1コンピュータからJavaクラスを動的にロードすることを特徴とする請求項13に記載の方法。

【請求項16】 透過的にストリーミングする際に、  
前記アプリケーションが実行されている間に前記モジュールセット内の第1モジュールを受信し、

前記第1モジュールを前記第2コンピュータにあるローカル記憶媒体上にストアし、

前記方法はさらに、

前記第1モジュールを前記アプリケーションに統合することを特徴とする請求項1に記載の方法。

【請求項17】 前記第1モジュールを統合する際に、前記第1モジュール内に含まれるロジックを前記アプリケーション内に含まれるロジックに統合することを特徴とする請求項16に記載の方法。

【請求項18】 前記アプリケーションは割込みステートメントを備え、

前記第1モジュールに含まれるロジックは、実行可能な置換コードを備え、

前記第1モジュールを統合する際に、

前記割込みステートメントを実行し、

実行プログラムを呼び出し、

前記割込みステートメントを前記置換コードで置き換え、

前記置換コードを実行することを特徴とする請求項17に記載の方法。

【請求項19】 前記アプリケーションは第1スタブ手続きを備え、

前記モジュールセットは第1コードシーケンスを備え、

前記モジュールセットを統合する際に、前記第1スタブ手続きをオペラティブに前記第1コードシーケンスで置き換えることを特徴とする請求項17に記載の方法。

【請求項20】 オペラティブに置き換える際に、アプリケーションから第1コードシーケンスにアプリケーション実行が移ることを可能にするように、第1スタブ手続きを変更することを特徴とする請求項19に記載の方法。

【請求項21】 コンピュータ可読メディア上に常駐するコンピュータプログラムであって、コンピュータに、  
アプリケーションと関連付けられたモジュールのコレクションにアクセスさせ

モジュール選択基準をストアするデータベースにアクセスさせ、

前記モジュール選択基準に従って前記コレクションからモジュールのシーケンスを選択することによってモジュールセットを形成させ、

前記モジュールセットを第2コンピュータに透過的にストリーミングさせる命令を備えることを特徴とするプログラム。

【請求項22】 コンピュータに透過的にストリーミングさせる前記命令は、前記コンピュータに、

前記コレクションから第1モジュールをリトリブさせ、

前記第1モジュールを前記第2コンピュータに送信させる命令をさらに備えることを特徴とする請求項21に記載のコンピュータプログラム。

【請求項23】 コンピュータ読取可能媒体上に常駐するコンピュータプログラムであって、コンピュータに、

アプリケーションを実行させ、

前記実行中のアプリケーションと関連付けられたモジュールを透過的に受信させ、

前記受信されたモジュールを前記実行中のアプリケーションとは独立にストアさせ、

前記受信されたモジュールを前記実行中のアプリケーションに統合させる命令を備えることを特徴とするプログラム。

【請求項24】 第1コンピュータおよび第2コンピュータを備える、コンピュータ間で情報モジュールを転送するためのシステムであって、

1) 前記第1コンピュータは、

a. アプリケーションを実行するための手段と、

b. 前記アプリケーションが動作している間に、前記アプリケーションと関連付けられたモジュールのシーケンスを受信するための手段と、

c. 前記受信されたシーケンス内の第1モジュールを前記アプリケーションに統合するための手段とを備え、

2) 前記第2コンピュータは、

a. 前記アプリケーションと関連付けられたモジュールのコレクションをストアするための手段と、

b. 前記コレクションからモジュールのシーケンスを選択するための手段と、

c. 前記第1コンピュータから前記第2コンピュータに選択された前記シーケンスを転送するための手段と

を備えることを特徴とするシステム。

【発明の詳細な説明】

【0001】

(背景情報)

クライアントサーバ環境で、クライアントコンピュータは、サーバと通信を行ってそのサーバにストアされた情報に遠隔アクセスすることができる。サーバとクライアントコンピュータとの間での情報の転送は、標準プロトコルおよびソフトウェアを使用して提供することができる。例えば、クライアントコンピュータにあるハイパーテキストマークアップ言語 (HTML) ブラウザアプリケーションが、HTTPサーバからウェブページを受信するのに、TCP/IPおよびハイパーテキストトランスファプロトコル (HTTP) を使用して、公衆インターネットを介して通信することができる。ウェブページは、フォーマットされたテキスト、ならびに埋め込まれたグラフィックスやサウンドなどのマルチメディアエレメントを含み得る。これらのマルチメディアエレメントが、クライアントによってダウンロードされて、ブラウザアプリケーションまたは「プラグイン」ブラウザコンポーネントによってユーザに提示されるということが可能である。例としてのブラウザアプリケーションは、ネットスケープナビゲータ (Netscape Navigator) 4.0 (登録商標) およびマイクロソフトインターネットエクスプローラ (Microsoft Internet Explorer) 4.0 (商標) を含む。

【0002】

クライアントコンピュータで使用されるブラウザアプリケーションは、ストリーミング (streaming) データ伝送プロトコルを使用してオーディオ情報およびビデオ情報を受信するのに、プラグインソフトウェアを使用することができる。ストリーミングプロトコルは、情報が受信されるにつれて、それがクライアントコンピュータによって提示されるということを可能にする。例えば、フルモーションビデオを、サーバからクライアントにフレームの線形ストリームとして送信することができる。クライアントに各フレームが着信するごとに、それがリアルタイムのフルモーションビデオ表示を生成するように表示され得る。オーディオストリーミングおよびビデオストリーミングは、クライアントが、クライアン

トアプリケーションにすべてのストリームが着信するのを待つことなく、情報を提示することを可能にする。オーディオストリーミングおよびビデオストリーミングは、例えば、リアルネットワーク社 (RealNetwork, Inc.) からのリアルオーディオ (RealAudio) (登録商標) およびリアルビデオ (RealVideo) (商標) によって提供される。

【0003】

また、ブラウザアプリケーションは、HTMLベースのウェブページの外観を向上させるのに、実行可能なソフトウェアアプレットを使用することができる。アプレットは、クライアントからの要求に応じてサーバからクライアントに送信されるソフトウェアプログラムである。代表的なアプレット使用では、HTMLベースのウェブページが、ブラウザアプリケーションにアプレットを要求させ、そのアプレットの実行を開始させるHTTPコマンドを含んでいる。したがって、アプレットは、ユーザと対話してデータの収集や処理を行うことができ、ネットワークを介してデータを通信することができ、結果をコンピュータ出力デバイス上に表示することができる。アプレットは、クライアントコンピュータにあるブラウザアプリケーションによって提供されるランタイム環境で実行するプログラミング言語で構築することができる。例えば、サンマイクロシステム社からのJava (登録商標) プログラミング言語は、Javaアプレットをウェブサーバにストアし、Javaインタプリタによる実行のためにウェブページに添付することができるようにする。Javaアプレットは、複数のJavaクラスで形成され得る。Javaクラスは、動的に生成された、そのクラスを実行する要求 (モジュール実行要求) に応答して、サーバからダウンロードすることができる実行可能なJavaコードを含む。実行中のアプレットがあるクラスによって提供される機能にアクセスしようとしたとき、そのJavaクラスがJavaインタプリタで利用可能ではない場合、Javaインタプリタは、サーバから動的にそのクラスをリトリブすることができる。マイクロソフト Visual Basic (登録商標)、またはマイクロソフト Visual C++ (登録商標) などの、他のプログラミング言語もまた、マイクロソフト ActiveX (商標) コントロールなど、アプレット様式のソフトウェアモジュールを



作成するのに使用することができる。

【0004】

また、ダウンロード可能なアプレットを、大きく複雑なプログラムを開発するのに使用することができる。例えば、複雑な財務プログラムをアプレットのコレクションから構築することができる。こうした財務プログラムでは、ユーザから情報を集め、支払を計算し、利率を計算し、また印刷された報告書を生成するのに別々のアプレットを使用することができる。特定のプログラム機能がユーザによって要求されると、その要求された機能と関連するアプレットが、サーバからリトリブされ得る。ただし、ソフトウェアアプリケーションのサイズが増大するにつれて、ネットワークを介してそのモジュールをリトリブするのに関連する遅延も同様に増大して、エンドユーザにとって許容できないものとなり得る。したがって、コンピュータ間でのソフトウェアの伝送の改善が望まれる。

【0005】

(概要)

本発明は、第1コンピュータと第2コンピュータとの間で、データモジュールをストリームとして伝送するための方法およびシステムを含む。モジュールは、そのモジュール間に「自然な」順序が存在するかどうかに関わらず、ストリームとして伝送することができる。例えば、データストリーム内容を判断するのに、データの自然な線形順序を頼りにするストリーミングアプリケーションとは異なり、開示のストリーミングメカニズムは、線形データ順序に従って動作するように拘束されていない。代わりに、ストリームとして伝送されるデータモジュールは、特定のデータ内容とは独立していることが可能な予め定められた基準を使用して、選択される。

【0006】

例としての適用では、開示のストリーミングメカニズムは、ソフトウェアモジュールのユーザ依存ストリーミングを提供することができる。例えば、ホームバンキングアプリケーションが、モジュール#1から#5を含み得る。第1バンキングアプリケーションユーザは、メニュースクリーンでのユーザ自身の入力選択に基づいて、1-3-4-5の順序でモジュールにアクセスすることができ、第

2ユーザは、2-4-1の順序でモジュールにアクセスすることができる。こうしたバンキングアプリケーションの場合、ストリーミングシーケンスを決定するために使用される予め定められた基準が、各ユーザのモジュール使用パターンを詳述することができる。アプリケーションのユーザと関連付けられた予め定められた基準は、第1ユーザがこのバンキングアプリケーションにアクセスしているとき、好ましいストリーミングシーケンス1-3-4-5を指定するが、第2ユーザがアプリケーションにアクセスしているときには、好ましいシーケンス2-4-1を指定することができる。したがって、ストリームとして伝送されるシーケンスは、履歴上のユーザ依存アクセスパターンに適合することが可能である。他のタイプの予め定められた基準もまた、使用することができる。開示のストリーミングメカニズムはまた、ハイパーテキストマークアップ言語データ、バイナリグラフィックス、およびテキストなどの非実行データ (non-executable data) をストリームとして伝送するのにも使用することができる。

#### 【0007】

一般的に一態様では、本発明は、第1コンピュータから第2コンピュータにモジュールを伝送するコンピュータ実装方法を備えている。第1コンピュータで、利用可能なモジュールのコレクションからモジュールのシーケンスを選択することによって、モジュールセットが形成される。選択されたモジュールのそれぞれは、第2コンピュータで動作しているアプリケーションと関連付けられる。選択されたモジュールは、第1コンピュータから第2コンピュータに、透過的に (transparently) ストリーミングすることができる。モジュールの選択は、予め定められた選択基準に従って行われ、第2コンピュータの実行環境とは独立している。

#### 【0008】

本発明の実装は、次の特徴のうちの1つまたは2つ以上を含むことが可能である。モジュールは、ハイパーテキストマークアップ言語、および/またはプログラムコードなどの、非実行データを含み得る。選択基準は、ストリーミング制御データベースにストアすることができる。ストリーミング制御データベースは、加重された値をコレクション内の選択されたモジュールの間での遷移と関連付け

る遷移レコード (transition record) を含むことができる。遷移レコード情報の処理は、パス決定アルゴリズムを使用することなどによって、モジュールのシーケンスを決定するのに使用することができる。ストリーミング制御データベースは、それぞれがモジュールの予め定められたシーケンスを識別するリスト記録を含むことができる。モジュールの選択は、リスト記録を選択することによって行うことができる。モジュールのシーケンスを選択することは、第2コンピュータから第1コンピュータにデータを送信して、シーケンス内の各モジュールを識別する、または実行中のアプリケーションのステータスを識別することを含み得る。例えば、ステータスを識別するデータは、一連のユーザ入力値を含み得る。

・【0009】

また、実装は、次の特徴のうちの1つまたは2つ以上を含み得る。モジュールセットのストリーミングには、割込みが行われることが可能で、第2シーケンスが決定され、この第2シーケンスのストリーミングが発生することが有り得る。モジュールセットのストリーミングには、第2コンピュータから第1コンピュータに送信される、特定のモジュールに対する要求によって割込みが行われ得る。例えば、Javaアプレットが、既にストリームとして第2コンピュータに伝送されたものではないJavaクラスにアクセスしようとすることによって、Javaクラスのストリームに割込みを行うことが有り得る。モジュールのシーケンスのストリーミングが行われ、実行中のアプリケーションとは独立に第2コンピュータにストアされ得る。つまり、実行中のアプリケーションは、ストリーミングを開始する必要がなく、ストリーミングに気付いている必要がない。ストリーミングされたモジュールはその後、ストリーミングされたモジュール内のロジックとアプリケーション内のロジックを相互接続することによって、第2コンピュータにあるアプリケーションに統合することができる。

【0010】

また、実装は、次の特徴のうちの1つまたは2つ以上を含み得る。アプリケーションは、割込みステートメントを含み得る。割込みステートメントの実行は、制御を実行プログラムに引き渡し得る。実行 (エグゼキュタ) プログラムは、割込みステートメントに応答し、実行中アプリケーションの処理の永久停止 (終了

)を妨げることによって、プログラムコードデバッガの方式で機能する。この後、実行プログラムは、割込みステートメントを(一般的に、置換ロジックのブロックの一部として)ストリーミングされたモジュールからの置換ロジックによって置き換えることによって、ストリーミングされたモジュール内のロジックをアプリケーションのロジックに統合することができる。この後、アプリケーションは、一般的に、割込みステートメントと入れ替えられた置換ロジックを実行することによって、動作を継続する。また、アプリケーションは、ストリーミングされたモジュール内のロジックによって置き換えることができるスタブプロシージャ(procedure)を含み得る。スタブプロシージャの置き換えは、スタブプロシージャコードを除去して、ストリーミングされたモジュールからのロジックでそれを置き換えることによるなど、直接的であることも、またはストリーミングされたモジュール内のロジックに対するリンクを作成することによるなど、オペラティブであることも可能である。

#### 【0011】

一般的に別の態様では、本発明は、コンピュータ読取可能媒体上に常駐するコンピュータプログラムを特徴とする。このコンピュータプログラムは、コンピュータに、アプリケーションと関連付けられたモジュールのコレクションにアクセスさせ、モジュール選択基準をストアするデータベースにアクセスさせ、そのモジュール選択基準に従ってそのコレクションからモジュールのシーケンスを選択することによってモジュールセットを形成させ、このモジュールセットを透過的に第2コンピュータにストリーミングさせるための命令を含んでいる。また、プログラムの実装は、コンピュータに、コレクションから第1モジュールをリトリブさせ、この第1モジュールを第2コンピュータに送信させるための命令も含み得る。

#### 【0012】

一般的に別の態様では、本発明は、コンピュータ読取可能媒体上に常駐するコンピュータプログラムを特徴とする。このプログラムは、コンピュータに、アプリケーションを実行させ、実行中のアプリケーションと関連付けられたモジュールを透過的に受信させ、この受信されたモジュールを実行中のアプリケーション

に統合させるための命令を含んでいる。

【0013】

一般的に別の態様では、本発明は、コンピュータ間で情報モジュールを転送するためのシステムを備えている。このシステムは、第1コンピュータおよび第2コンピュータを含む。第1コンピュータは、アプリケーションを実行するための手段、アプリケーションが実行されている間にそのアプリケーションと関連付けられたモジュールのシーケンスを受信するための手段、および受信されたシーケンス内の第1モジュールをアプリケーションに統合するための手段を含む。第2コンピュータは、アプリケーションと関連付けられたモジュールのコレクションをストアするための手段、このコレクションからモジュールのシーケンスを選択するための手段、およびこの選択されたシーケンスを第1コンピュータから第2コンピュータに転送するための手段を含む。

【0014】

また実装は、次の利点のうちの1つまたは2つ以上を含む。アプリケーション、コードモジュール、またはデータモジュールをダウンロードしているときに経験する遅延を短縮することができる。ソフトウェアおよびデータモジュールを、特定のエンドユーザの要求に従って、クライアントワークステーションに予測的に配信することができる。モジュールがサーバからクライアントにストリームとして伝送される順序を、動的に決定することができる。モジュール配信シーケンスのコレクションを特定のアプリケーションまたはユーザと関連付けることができ、また、そのシーケンスを動的に更新することができる。モジュール配信シーケンスを、ソフトウェア使用の個別パターン、またはモジュール使用と関連付けられた、ストアされている統計に基づいて決定することができる。アプリケーションの実行の間に、モジュールストリーミングに割り込みを行い、それを変更することができる。実装は、以下の記載と特許請求の範囲から明らかになるとおり、追加または代替の利点を含み得る。

【0015】

(詳細な説明)

図1を参照すると、ワイドエリアネットワーク100が示されている。ネット

ワーク100において、クライアントコンピュータ101は、リンク103および104上でデータネットワーク130にデータを送ることによって、サーバコンピュータ102と通信することができる。データネットワーク130は、クライアント101とサーバ102との間でデータを送ることができる複数のノード131~134を含むことができる。クライアントコンピュータ101は、TCP/IP、HTTP、および他のプロトコルを使用して、データを送信し、受信することができる。例えば、クライアント101は、HTTPを使用して、サーバ102からウェブページを要求することができる。

#### 【0016】

サーバ102からクライアント101に送られたウェブページおよびマルチメディアデータは、それらに関連付けられた自然線形シーケンスを有することができる。ビデオデータの自然シーケンスは、ビデオフレームの線形な順序であり、一方テキストの自然シーケンスは、テキストのページがドキュメント内で構成されている順序である。自然線形シーケンスを有するデータは、サーバからクライアントにストリームし、ダウンロードの遅延を最小にすることができる。ストリーミングシステムでは、線形シーケンスのより前のアイテムを処理および/または表示しながら、後続のアイテムをクライアントコンピュータにダウンロードすることができる。アイテムの処理および/または表示が完了したとき、処理または表示または完全に受信した「ストリーミングされた」アイテムが、直ちに開始することができる。アイテムが要求されるとき、ストリーミングされたアイテムの受信は、完全または部分的に終了しているので、そのストリームされたアイテムを要求するユーザまたはクライアントアプリケーションは、低減されたダウンロードの遅延を知覚することになる。例えば、ドキュメントの第1ページをユーザが検索する場合、第1ページを読みながら第2ページをダウンロードすることができる。ユーザがドキュメントの第2ページを読み続ける場合、ハードディスクドライブ上のキャッシュエリア内などのクライアントのところでページは利用可能となり、追加のダウンロードの遅延なしで読むことができる。

#### 【0017】

ソフトウェアの実行は、予想可能な自然線形順序に従わない可能性がある。ソ

ソフトウェアは、ジャンプ命令文、ブレイク命令文、プロシージャコール、および実行コードのセクション内で突然実行を転送させる他のプログラミング構造体を含むことができる。相互に関係付けられたコードモジュール（コードセグメント、コードクラス、アプレット、プロシージャ、およびコードライブラリなど）の処理中にトラバース（traverse）された実行パスは、しばしば非線形で、ユーザに依存するようになり、アプリケーションプログラムを実行する度に変わることができ、また様々なデータアイテムの状態に応じて変わることができる。自然な順序は欠如している可能性があるが、モジュールをストリームする有利な順序を決定することができる。その順序は、コンピュータの内部構造または内部オペレーティングシステム（実行環境）の考慮事項に依存しない基準を用いて決定することができる。

#### 【0018】

図2を参照すると、ソフトウェアアプリケーション200は、「A」から「H」の複数のモジュールを含むことができる。モジュール「A」から「H」は、Javaクラス、C++プロシージャライブラリ、またはサーバのところでストアすることができる他のコードモジュールとすることができる。またモジュール「A」から「H」のいくつかは、ハードディスクキャッシュ内やクライアントコンピュータにストアされているソフトウェアライブラリの部分として、クライアントコンピュータのところでストアすることができる。クライアントコンピュータがアプリケーション200の実行を開始するとき、モジュール「A」などの第1モジュールは、サーバからダウンロードすることができ、クライアント410のところで実行を開始することができる。モジュール「A」が処理されているとき、その中に含まれているプログラミング命令文は、例えばモジュール「E」にランチすることができる。モジュール「E」がまだクライアントのところに存在しない場合、モジュール「A」の実行を中断し、モジュール「E」をサーバから検索（リトリブ）することができ、次いでモジュール「E」のコードの実行を開始することができる。そのようなシナリオでは、ユーザは、サーバからモジュール「E」をリトリブすることに関連付けられたモジュールダウンロードの遅延を経験することになる。

#### 【0019】

ユーザが経験するモジュールダウンロードの遅延を最小にするために、モジュール「E」を、透過的にサーバからクライアントコンピュータにストリームすることができる。透過的なストリーミングにより、他の相互に関係付けられたモジュール「A」を実行しながら、将来のモジュールの使用を予測し、モジュールをダウンロードすることが可能となる。図4を参照すると、透過的なストリーミングを提供する例示的ソフトウェア構造400が示されている。ソフトウェア構造400は、ストアされたソフトウェアモジュールのデータベース403を有するサーバ401を含む。サーバ401は、通信リンク上で、透過的にソフトウェアモジュール405のストリームをクライアントコンピュータ410に送信することができる。通信リンクは、アナログモデム接続、デジタル加入者ライン接続、ローカルエリアネットワーク接続、またはサーバ401とクライアント410の間のその他のタイプのデータ接続とすることができる。特定のソフトウェアモジュールがクライアント410のところで実行されているとき、追加のモジュールが、サーバ401からクライアント410に送られる。動的なストリーミングの実装では、サーバとクライアントの間でモジュールをストリームする順序は、使用されている特定のクライアントコンピュータ410に基づいて、クライアントコンピュータのユーザに基づいて、および他の動的に決定される因子に基づいて変更することができる。

#### 【0020】

図3を参照すると、アプリケーションモジュール「A」から「H」の実行順序は、モジュールの線形シーケンスではなく、有向グラフ300に類似することができる。例えば、グラフ300で示すように、モジュール「A」を実行した後、モジュール「B」、「D」、または「E」のところで実行を続けることができる。モジュール「B」を実行した後、モジュール「C」または「G」のところで実行を続けることができる。続いて、実行パスは、追加のモジュールへと進み、以前に実行したモジュールに戻ることができる。

#### 【0021】

サーバ401は、ストリーム制御情報402を使用して、サーバ401からク



クライアント410にモジュールをストリームする順序を決定することができる。例えば、ストリーム制御情報402は、有向グラフ300によって表されているようなソフトウェアモジュール間の予測された実行フローを含むことができる。ダウンロードしたモジュールをクライアント410によって実行するとき、クライアントは、制御データ415をサーバ401に送り、モジュールをサーバ401からクライアント410にストリームする順序を、動的にアップデートおよび変更することができる。制御データ415を使用して、サーバ401から特定のモジュールを要求し、アプリケーションプログラムの現在の実行状態に関するデータを送り、クライアントのローカル記憶装置411にあるモジュールの現在の目録(inventory)を詳述し、およびユーザの入力選択と、プログラム実行の統計と、クライアントコンピュータ410およびその実行ソフトウェアに関して得られた他のデータとを報告することができる。

#### 【0022】

ストリーム405内で、サーバ401からクライアント410に送られたモジュールのシーケンスは、ストリーム制御ファイル402を用いて決定することができる。ストリーム制御ファイル402は、クライアント410のところで必要となるモジュールを予測するために、サーバが使用するデータを含むことができる。グラフをベースとする実装では、制御ファイル402は、有向グラフのノードとしてモジュールを表すことができる。また制御ファイル402は、モジュール間の可能な実行の移動を、ノードを相互接続する頂点(「エッジ(edge)」)として表すことができる。表1を参照すると、重み付きのグラフの実装では、ストリーム制御ファイル402は、モジュール間の可能な移動を表す頂点のリストを含むことができる。例えば、表1は、グラフ300(図3)の「A」から「H」のモジュール間の全ての可能な移動を表す頂点を列挙している。表1の各頂点は、モジュール間で特定の遷移が生じる相対的な可能性を示す重みの値を含む。表1の例では、重みの値がより大きいと、移動の可能性がより低いことを示す。サーバ401は、最短パスのグラフ・トラバース・アルゴリズム(graph traversal algorithm)(また「最低コスト」アルゴリズムとしても知られている)を適用して、現在実行しているモジュールに基づいて、所望のモジュールストリー

ミングシーケンスを決定することができる。最短パスアルゴリズムの例は、1987年のMischa Schwartz、Addison WesleyによるTelecommunications Networks: Protocols, Modeling and Analysisの§6に見出すことができる。

【0023】

【表1】

表1：グラフエッジ表

エッジ	重み
(A, B)	1
(A, D)	7
(A, E)	3
(B, C)	1
(B, G)	3
(C, E)	2
(C, G)	6
(D, F)	2
(E, H)	2
(F, H)	1
(G, E)	3
(G, H)	4

【0024】

例えば、表2は、モジュール「A」と表1の残りのモジュールの間の最小パスの重みを表す。

【0025】

【表2】

表2：アプリケーション「A」からの最短パス

から	へ	最短パスの重み	パス
A	B	1	A-B
	C	2	A-B-C
	D	7	A-D
	E	3	A-E
	F	9	A-D-F
	G	4	A-B-G
	H	5	A-E-H

【0026】

表2に示す重みの値に基づいて、サーバ401は、モジュール「A」の実行中に、モジュールストリーミングシーケンス「B」、「C」、「E」、「G」、「H」、「D」、「F」が有利であることを決定することができる。制御データ415によって報告された可能性があるように、決定されたシーケンス内の特定のモジュールが既にクライアント402のところに存在する場合は、サーバ401は、モジュール405のストリームからそのモジュールを削除することができる。シーケンス「B」、「C」、「E」、「G」、「H」、「D」、「F」の実行中に、モジュール「A」の実行が完了し、他のモジュールの実行が開始する場合、サーバは、シーケンス「B」、「C」、「E」、「G」、「H」、「D」、「F」の引渡しに割り込み、現時点の実行モジュールに基づいて新しいシーケンスを計算し、新しく計算したストリーミングシーケンスに基づいてストリーミングを再開することができる。例えば、実行がモジュール「A」からモジュール「B」に移動する場合、モジュール「B」が現在実行しているモジュールであることを示す制御データ415を、クライアント410からサーバ401に送ることができる。モジュール「B」がまだクライアント401のところで利用可能でない場合は、サーバ401は、クライアントへのモジュール「B」の引渡しを終了し、新しいモジュールストリーミングシーケンスを決定することになる。最短パスのル

ーディングアルゴリズムを、開始点とするモジュール「B」に基づいて、表1のエッジに適用することにより、表3に示すように、モジュール「B」とグラフ300（図3）の他のモジュールの間の最小パスの重みを決定することができる。

【0027】

【表3】

表3：モジュール「B」からの最短パス

から	へ	最短パスの重み	パス
B	C	1	B-C
	E	5	B-C-E
	G	3	B-G
	H	7	B-C-E-H

【0028】

表3に示す最短パスの重みに基づいて、サーバ401は、モジュールストリーミングシーケンス「C」、「G」、「E」および「H」が有利であることを決定することができる。

【0029】

また、他のアルゴリズムを使用して、モジュールのストリーミングシーケンスを決定することができる。例えば、より重い重みの付いたエッジが、グラフに示すモジュール間の好ましいパスであることを示している重み付きのグラフ300を使用することができる。表4では、より大きな値を割り当てられた重みの値は、モジュール間の好ましい移動を示す。例えば、エッジ（A、B）、（A、D）、および（A、E）は、モジュールAからの3つの可能な移動である。エッジ（A、B）は、エッジ（A、D）および（A、E）より大きな重みの値を有しているので、好ましく、したがって、モジュール「A」が開始点である場合、モジュール「D」または「E」の前にモジュール「B」をストリームすることが好ましい可能性がある。例えば、エッジの重みの値は、特定のモジュールがクライアントによって要求された回数の履歴カウント、コードモジュールの相対送信時間、

またはシステムアドミニストレータによって経験的に決定され、サーバ401で表402にストアされている値、とすることができる。他のエッジの重みの計算方法を使用することもできる。

【0030】

【表4】

表4： 好ましいパスの表

エッジ	重み
(A, B)	100
(A, D)	15
(A, E)	35
(B, C)	100
(B, G)	35
(C, E)	50
(C, G)	20
(D, F)	50
(E, H)	50
(F, H)	100
(G, E)	35
(G, H)	25

【0031】

好ましいパス（重く重み付けしたエッジを第1とする）の実装では、より大きな重みの値を有しているグラフ300のエッジが好まれる。以下の例示的アルゴリズムを使用して、好ましいパスの実装でのモジュールストリーミングシーケンスを決定することができる。

【0032】

1：2つの空の順序付けしたセットを作成する：

i) 「S」がノード識別子であり、「W」がノード「S」に到達するように

トラバースすることができるエッジの重みである対  $(S, W)$  をストアする候補セット

i i) 決定されたコードモジュールのストリームをストアするストリームセット

2:  $S_i$  を開始ノードとする。

3: ノード  $S_i$  をストリームセットに添付し、候補セットから任意の対  $(S_j, W)$  を除去する。

4: 重み  $W_j$  を有するエッジ  $(S_i, S_j)$  によってノード  $S_i$  から到達することができる各ノード  $S_j$  に対し:

{

$S_j$  がストリームセットのメンバでない場合、候補セットに  $(S_j, W_j)$  を追加する。

$S_j$  が候補セットの複数の対に現われる場合、最大重み  $(S_j, W)$  の対を除いて、全ての対を候補セットから除去する。

}

5: 候補セットが空でない場合、

候補セットから最大重みの対  $(S_k, W_k)$  を選択する。

$S_i = S_k$  とする。

ステップ3で反復する。

【0033】

例えば、表5に示すように、ノード「A」のところで開始し、上記のアルゴリズムを表4のエッジに提供することにより、ストリームセット {A、B、C、E、H、G、D、F} が作成される。

【0034】

【表5】

表5：ストリームセットの計算

繰返し	{ストリームセット} / {候補セット}
1	{A}/{(B,100)(D,15)(E,35)}
2	{A,B}/{(D,15)(E,35)(C,100)(G,35)}
3	{A,B,C}/{(D,15)(E,35)(G,35)}
4	{A,B,C,E}/{(D,15)(G,35)(H,50)}
5	{A,B,C,E,H}/{(D,15)(G,35)}
6	{A,B,C,E,H,G}/{(D,15)}
7	{A,B,C,E,H,G,D}/{(F,50)}
8	{A,B,C,E,H,G,D,F}/{}

## 【0035】

実装は、代替のアルゴリズムを選択して、ストリームセットを計算するすることができる。

## 【0036】

また、アプリケーションストリーミングを用いて、アプリケーションまたはモジュールのサブセクションをストリームすることができる。例えば、C、C++、フォートラン、パスカル、またはアセンブリ言語で書かれているアプリケーションなどのコンパイルされたアプリケーションのサブセクションは、サーバ401からクライアント410にストリームすることができる。図5Aを参照すると、アプリケーション500は、メインコードモジュール501およびコードライブラリ510ならびに515などの複数のコードモジュールを含むことができる。メインモジュール501は、アプリケーションが開始されるとき実行されるプログラムコードを含む。コードライブラリ510および515は、ヘッダデータ511および516、並びに、メインモジュール501および他のライブラリプロシージャから直接または間接的に呼び出される実行可能なプロシージャ512～514および517～519を含むことができる。

## 【0037】

マイクロソフトウィンドウズ95/マイクロソフトビジュアルC++の実装では、メインコードモジュール501は、コンパイルされたC++「メイン」プロシージャを含むことができ、ライブラリモジュール510および515は、コンパイルされたC++オブジェクトコードプロシージャを有する動的リンクライブラリとすることができる。ヘッダデータ511および516は、動的にライブラリ510および515をメインモジュール510とリンクするために、オペレーティングシステムのリンクプロシージャが使用する記号名を含むことができる。またヘッダデータは、ライブラリ内の各プロシージャの位置 (location) を示すことができる。ジャンプの表の実装では、コーリングプロシージャは、ヘッダ511または516内の予め定められた位置にジャンプし、そこから追加コードおよび/またはデータにアクセスし、結果的に続いてプロシージャの開始にジャンプすることによって、ライブラリプロシージャ512~514、517~519にアクセスすることができる。

#### 【0038】

アプリケーションのコードモジュールおよびライブラリ内のデータおよびプロシージャは、何百または何千というバイトの長さとすることができる。アプリケーションを実行する前に、クライアントは、モジュールおよびライブラリの長いセットの検索を必要とすることがある。モジュールおよびライブラリのセットのサイズを低減することによって、アプリケーション実行前に経験された当初の遅延を低減することができる。アプリケーション500のストリーミングの実装において、アプリケーションのコードモジュールのサブセクション内のコードを除去し、短縮したストリーミングの「スタブ」プロシージャで置き換えることができる。アプリケーションコードをストリーミングスタブプロシージャで置き換えることにより、モジュールサイズおよび関連付けられた送信の遅延を低減することができる。例えば、図5Aおよび5Bを参照すると、コードライブラリ510は、長さが4キロバイト (Kバイト) であるヘッダ511と、それぞれ32Kバイト、16Kバイト、および8Kバイトであるプロシージャ512~514を含むことができる。図5Bおよび5Cを参照すると、ライブラリ510のサイズを低減するために、プロシージャコード512~514をライブラリ510から除



去し、サーバ401 (図4) のところでストリーミングコードモジュールデータベース403にストアすることができる。除去されたプロシージャコード512～514は、「スタブ」プロシージャ515～517によって置き換えられ、結果的に、ライブラリ510の代わりに、アプリケーションモジュール501および520とリンクすることができる、サイズが低減したコードライブラリ530を得る。ライブラリ530のヘッダデータ511は、アップデートされたジャンプ、またはスタブプロシージャ515～517がプロシージャ512～514のリンクタイム代用物として作用することを可能とするリンク情報を含むことができる。

#### 【0039】

サーバ401は、メインモジュール501と、ライブラリモジュール520と、「ストリームされた」ライブラリ530とを送ることによって、クライアント410にアプリケーション500のストリーミング可能なバージョンを提供することができる。いくつかの実装では、アプリケーション500に対する要求に応答して、クライアント410に、ストリーミングサポートファイル535を提供することができる。ストリーミングサポートファイル535は、サーバ401とクライアント410との間で、コードストリーミングを容易にするために、スタブ515～517によってアクセスされるプロシージャを含む。クライアント410のところで、モジュール501、520、530、および535をリンクし、結果的に得られるアプリケーションの実行を開始することができる。メインモジュール501および様々な呼び出されたプロシージャをクライアント410のところで実行するとき、データベース403にストアされたコードモジュールを、サーバ401からクライアント410にストリームすることができる。データをストリーム403に含め、ストリームされたコードモジュールに関連付けられたスタブプロシージャ414～417を識別することができる。ストリームされたモジュールがクライアントのところで受信されるとき、それらは、実行アプリケーションと統合される。

#### 【0040】

添付されたモジュールの実装では、ストリームされたコードモジュールは、受

信したモジュールを対応するライブラリまたはコードファイルに添付することによって、実行アプリケーションと統合される。例えば、図5 Cおよび5 Dを参照すると、モジュール5 1 2～5 1 4がサーバからクライアントにストリームされるとき、それらはライブラリファイル5 3 0に添付され、それにより、増大されたライブラリファイル5 4 0を形成する。モジュール5 1 2～5 1 4をサーバ4 0 1からストリームし、ファイル5 3 0に添付するとき、ヘッダデータ5 1 1またはスタブデータ5 1 5～5 1 6はアップデートされ、したがって、現時点で添付されたモジュールは、呼び出し（コーリング）プロシージャからアクセスすることができる。例えば、図5 Dを参照すると、追加の「ジャンプ」を、各スタブプロシージャ5 1 5～5 1 7とそれに関連付けられた添付のモジュール5 1 2～5 1 4の間で追加することができる。代替として、ヘッダデータ5 1 1をアップデートし、スタブ5 1 5～5 1 7の代わりに、プロシージャ5 1 2～5 1 4をアクセス可能とすることができる。スタブ置換の実装では、モジュールがサーバ4 0 1から受信されるとき、スタブ5 1 5～5 1 6はプロシージャモジュール5 1 2～5 1 4で置き換えられる。スタブ置換は、置換コードが受信されるとき、コードモジュールまたはライブラリ内で、残りのスタブまたはプロシージャの位置を変更あるいは再構成することを要求する可能性がある。実装は、ストリームされたコードを実行アプリケーションおよびモジュールと統合するさらに他の方法を使用することができる。

#### 【0041】

いくつかのシナリオでは、上記の例ではスタブ5 1 5～5 1 7によって置き換えられているプロシージャコード5 1 2～5 1 4などの除去されたコードは、それをサーバ4 0 1からストリームし、モジュール5 3 0と統合する前に、必要とする可能性がある（他のプロシージャによって呼び出される）。そのような場合では、スタブコード5 1 5～5 1 6は、ストリーミングサポートライブラリ5 3 5内でストリーミング機能にアクセスし、必要なプロシージャを獲得することができる。そのために、ストリーミングサポートライブラリ5 3 5は、制御データ4 1 5をサーバ4 0 1に送り、必要なプロシージャを要求することができる。応答して、サーバ4 0 1は、現在のモジュールストリーム4 0 5を中止して、要求

されたモジュールを送ることができる。要求されたモジュールを受信する際に、ストリーミングサポートライブラリ 5 3 5 内のプロシージャを使用して、受信したモジュールをアプリケーションと統合し、要求されたモジュールの実行を続けることができる。その後サーバは、要求されたモジュールまたはクライアントから受信された他の制御データ 4 1 5 に基づいて、新しいモジュールストリームを決定することができる。

#### 【0042】

コードモジュールは、スタブプロシージャを使用せずに、サイズを低減することができる。例えば、図 4、5 A、5 B、および 5 E を参照すると、割込みが駆動された実装では、プロシージャコード 5 1 2 ~ 5 1 4 をコードライブラリ 5 1 0 から除去し、データベース 4 0 3 にストアすることができる。次いで、ヘッダ情報 5 1 1 並びに除去されたプロシージャコード 5 1 2 ~ 5 1 4 のサイズおよび位置を示すデータを、クライアント 4 1 0 に送信することができる。クライアント 4 1 0 は、除去されたプロシージャコード 5 1 2 ~ 5 1 4 の代わりに、一連の割込み命令文を添付することによって、新しいライブラリ 5 5 0 を構築することができる。アプリケーション 5 0 0 を実行するとき、コードライブラリ 5 5 0 をライブラリ 5 1 0 の代用とし、プログラム 5 0 0 の実行を開始することができる。プログラム 5 0 0 を実行するとき、除去されたプロシージャコード 5 1 2 ~ 5 1 4 をクライアント 4 1 0 にストリームし、ローカルデータベース 4 1 1 にストアする。アプリケーション 5 0 0 が、プロシージャコード 5 1 2 ~ 5 1 4 を実行することを試みる場合、代わりに、プロシージャコード 5 1 2 ~ 5 1 4 と置き換わっている割込み命令文の 1 つを実行することができる。停止 (halt) 命令文の実行は、プログラム 5 0 0 の実行を中止し、制御をストリーミングエグゼキュタプログラム 4 1 5 に転送する。

#### 【0043】

エグゼキュタ 4 1 5 は、従来のランタイムオブジェクトコードデバッガの技術と同様のインターフェース技術を実施し、それにより、エグゼキュタ 4 1 5 は、アプリケーション 5 0 0 によって生成された割込みをインターセプトし、処理することが可能となる。エグゼキュタ 4 1 5 によって割込みがインターセプトされ

るとき、クライアントの実行プラットフォーム（オペレーティングシステム）割込み処理機能の部分としてエグゼキュタ415に提供されたデータを使用して、割込みが実行されたモジュール550と、モジュール内の割込みコードのアドレスを識別することができる。次いでエグゼキュタ415は、割込み位置に関連付けられたプロシージャコード512～514が、クライアントに送られたモジュールストリーム415の部分として受信されたかを決定する。適切なプロシージャコードが受信されている場合、エグゼキュタ515は、識別した割込みを対応するコードで置き換える。例えば、プロシージャ512～514を、クライアント410にストリームされる4キロバイトのコードモジュールに分割することができる。割込み命令文が、アプリケーション500によって実行されるとき、エグゼキュタ415は、割込みをインターセプトし、割込み命令文を含む適切な4キロバイトのコードブロックを決定し、決定されたコードブロックを受信したコードモジュールで置き換える。適切なコードモジュールがまだ受信されていない場合、明確な要求をクライアント410からサーバ401に送り、コードモジュールを検索した後、ライブラリ550に挿入することができる。その後エグゼキュタ415は、遭遇した割込みのアドレスのところでアプリケーション500を再開することができる。

#### 【0044】

また実装は、モジュールまたはライブラリの全体をストリームすることができる。例えばコードライブラリ510および515をサーバ401からクライアント410にストリームする間、メインコードモジュール501をサーバ401から受信し、クライアント410のところで実行を開始することができる。ストリームされたモジュールを実行モジュールと統合することは、クライアント410の動的モジュールリンク機能（facility）によって提供することができる。例えば、マイクロソフトビジュアルC++6.0によって提供される遅延インポートローディングを使用して、ストリームされたモジュール510および515を、実行モジュール501と統合することができる。ストリームされたモジュールの動的リンクは、ストリームされたモジュールをローカルハードディスクドライブ上またはクライアント410のリンクローディング機能によってアクセス可能な

他の格納装置の位置にストアすることによって、容易にすることができる。例示的な実装では、クライアント410のオペレーティングシステムのリンク機能を変更することによって、ストリーミングが容易になり、したがって、モジュールがまだクライアント401にストリームされていない場合、リンク機能は、制御データ415をサーバ401に送ることができる。

#### 【0045】

保護メモリのコンピュータシステムでは、実行アプリケーションコードおよびデータの直接操作が限定されている可能性がある。そのようなシステムでは、ストリームされたモジュールを実行アプリケーションと統合することをサポートするために、「カーネル」レベルのプロセスまたはプロシージャを必要とする可能性がある。そのような場合、ストリーミングサポート535は、クライアントがアプリケーション500を要求する前に、クライアント410のところで、サポートプロシージャをインストールすることによって、予め準備することができる。

#### 【0046】

ストリームセットを決定する他の方法を使用することができる。リストをベースとする実装では、ストリーミング制御ファイルは、モジュールストリーミングシーケンスの予め定められたリストを含むことができる。例えば、ストリーミング制御ファイル402は、第1ユーザに関連付けられたモジュールストリーミングシーケンスのリストと、第2ユーザに関連付けられた第2モジュールストリーミングシーケンスのリストとを含むことができる。クライアント410からサーバ401に送られた制御データ415は、クライアント410のところで、現在のユーザを識別することができる。サーバに対しユーザが識別された後は、サーバは、ユーザの関連付けられたストリーミングシーケンスのリストに従って、ソフトウェアモジュールをストリームすることができる。ユーザベースのストリーミングデータは、ユーザの過去の行動を使用してそのユーザがアクセスするモジュールの順序を予測することができる場合に有利である可能性がある。

#### 【0047】

グラフベースのストリーミング制御ファイルの実装では、ノードを接続するエ

ッジの重みは、静的または動的に決定することができ、またこれまでの使用データの収集に基づいて決定することができる。例えば、プログラマ制御の実装では、ソフトウェアのプログラマは、プログラマのソフトウェアコードと予期されるアプリケーションの使用パターンの知識に基づいて、ノード間の特定の移動が生じる可能性を推定する。代替として、アプリケーションプロファイリングプログラムを使用して、様々なアプレット、クラス、またはコードモジュール間の移動を記録するランタイム実行データを収集し、それにより、特定の移動が生じる可能性を決定することができる。クライアントフィードバックの実装では、モジュール実行中に、クライアント410からサーバ401に送られた制御データ415を使用して、モジュール使用の統計的データベースを構築し、そのデータベースに基づいて、モジュールをストリーミングする順序を決定する。

#### 【0048】

クライアント制御のストリーミングの実装では、ストリーミング制御データ402をクライアント410に配置し、クライアント410からサーバ401に送られた制御データ415を使用して、連続してサーバからモジュールのストリームを要求することができる。例えば、クライアントコンピュータ410が第1モジュールを実行している間、バックグラウンドプロセスは、制御データ415をサーバに送り、クライアントコンピュータ410のハードディスク411上でバッファすることができる追加のモジュールを要求することが可能である。クライアント制御のストリーミングの実装は、存在するHTTPサーバおよびHTTPプロトコルを使用して、クライアント410からサーバ401に要求を送り、サーバ401からクライアント410にソフトウェアモジュールを送ることができる。さらに、ソフトウェアモジュールのストリーミングは、上記の説明で強調されているが、ハイパーテキストマークアップ言語、2進グラフィックファイル、およびテキストなどの非実行データは、モジュールの収集としてストリームすることができる。

#### 【0049】

実装は、「ハンドシェーキング」プロシージャを含むことができ、それにより、アプリケーション実行の開始時に、サーバ401とクライアント410の間で

制御データ415が送られる。ハンドシェーキングデータは、クライアントおよびサーバのところにあるアプリケーションモジュールの在庫を含むことができる。そのようなハンドシェーキングデータにより、クライアント410とサーバ401の両方が、それぞれのソフトウェアモジュールの在庫を決定し、その在庫情報に基づいて、ソフトウェアモジュールのストリームを最適化することが可能となる。

#### 【0050】

履歴（ヒストリ）に依存する実装では、サーバまたはクライアントは、モジュール間の一連の移動に関するデータをストアし、移動の履歴に基づいて、新しいモジュールのストリームを計算することができる。例えば、図3を参照すると、モジュール「G」がパスA-B-Gによって到達された場合、サーバまたはクライアントは、次が「H」であるモジュール「E」をストリームすることを決定することができる。他方、モジュール「G」がパスA-B-C-Gによって到達された場合、ストリーミングシーケンスは、モジュール「H」のみを含むことができる。

#### 【0051】

本発明は、コンピュータハードウェア、ファームウェア、ソフトウェア、デジタル電子回路において、またはそれらの組み合わせで実施することができる。本発明の装置は、プログラム可能なプロセッサで実行するためのマシン読取り可能記憶装置内に有形で組み込まれているコンピュータプログラム製品において実施することができる。また、本発明の方法のステップは、入力データに関して動作し、および出力を作成することによって本発明の機能を実施するために、命令プログラムを実行するプログラム可能なプロセッサによって実施することができる。

#### 【0052】

本発明は、データストアシステムからデータと命令を受信し、およびデータストアシステムにデータと命令を送信するために結合されている少なくとも1つのプログラム可能なプロセッサと、少なくとも1つの入力デバイスと、少なくとも1つの出力デバイスとを含むプログラム可能なシステム上で実行可能である1つ

または2つ以上のコンピュータで実施することが有利である可能性がある。各コンピュータプログラムは、高水準手続プログラミング言語またはオブジェクト指向のプログラミング言語、あるいは所望であればアセンブリ言語または機械言語で実施することができる。いずれの場合でも、言語は、コンパイルされた言語または解釈された言語とすることができる。例として、適切なプロセッサは、汎用および専用のマイクロプロセッサの両方を含む。一般に、プロセッサは、読取り専用メモリおよび／またはランダムアクセスメモリから命令およびデータを受信する。

#### 【0053】

有形に組み込んでいるコンピュータプログラムの命令およびデータに適したストレージデバイスは、例としてEPROM、EEPROM、およびフラッシュメモリデバイスなどの半導体メモリデバイス、内部ハードディスクおよび取外し可能ディスクなどの磁気ディスク、光磁気ディスク、およびCD-ROMディスクを含む、不揮発性メモリの全ての形態を含む。上記のいずれも、専用に設計されたASIC（特定用途向け集積回路）によって補足し、またはその内部に組み込むことができる。

#### 【図面の簡単な説明】

##### 【図1】

コンピュータネットワークを示す図である。

##### 【図2】

コンピュータソフトウェアアプリケーションモジュールを示す図である。

##### 【図3】

本発明による、有向グラフである。

##### 【図4】

本発明による、サーバおよびクライアントを示す図である。

##### 【図5A】

本発明による、アプリケーションコード構成要素を示す図である。

##### 【図5B】

本発明による、アプリケーションコード構成要素を示す図である。



**【図 5 C】**

本発明による、アプリケーションコード構成要素を示す図である。

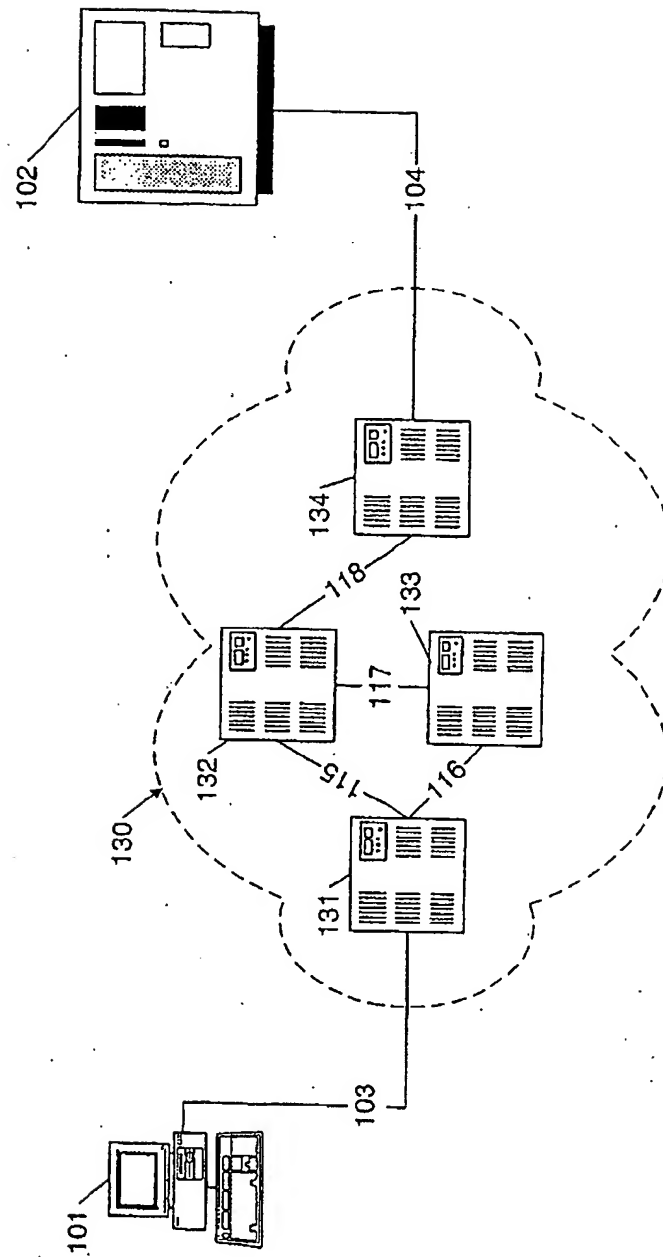
**【図 5 D】**

本発明による、アプリケーションコード構成要素を示す図である。

**【図 5 E】**

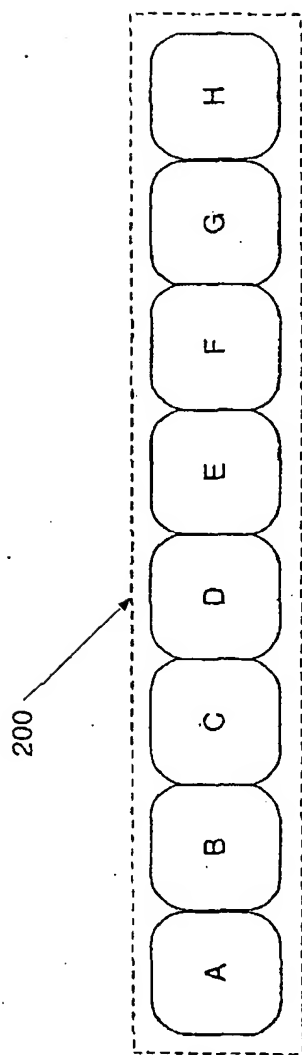
本発明による、アプリケーションコード構成要素を示す図である。

100

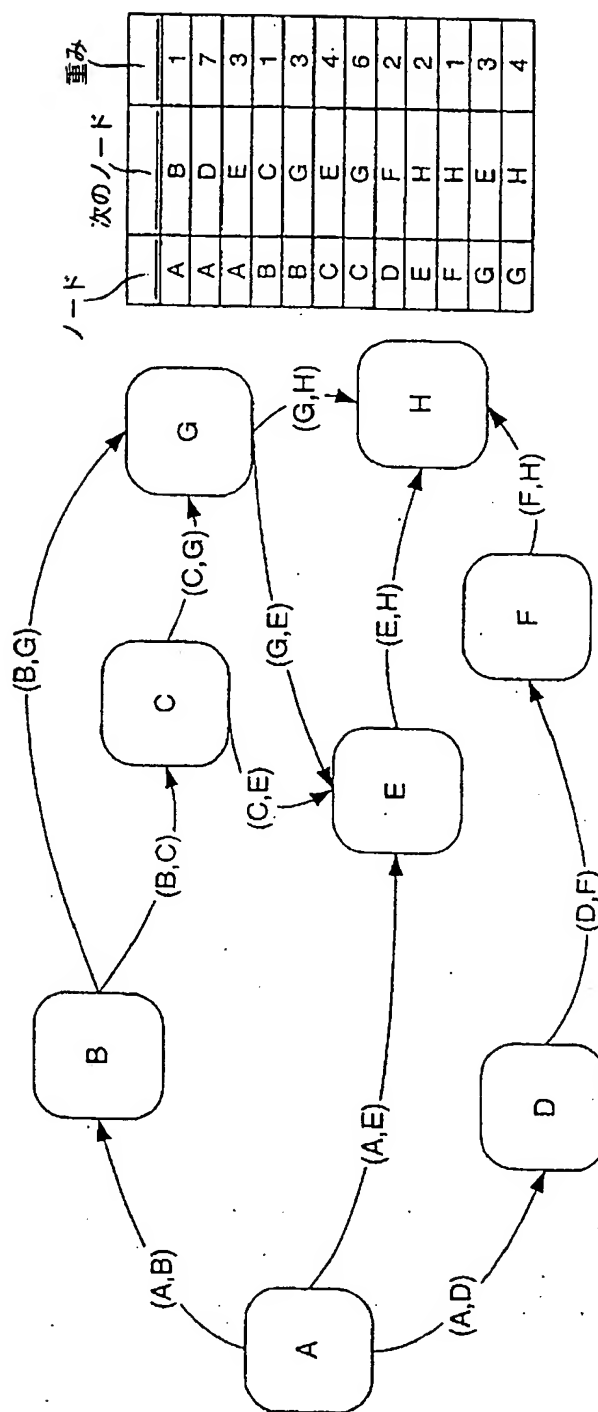


(従来の技術)

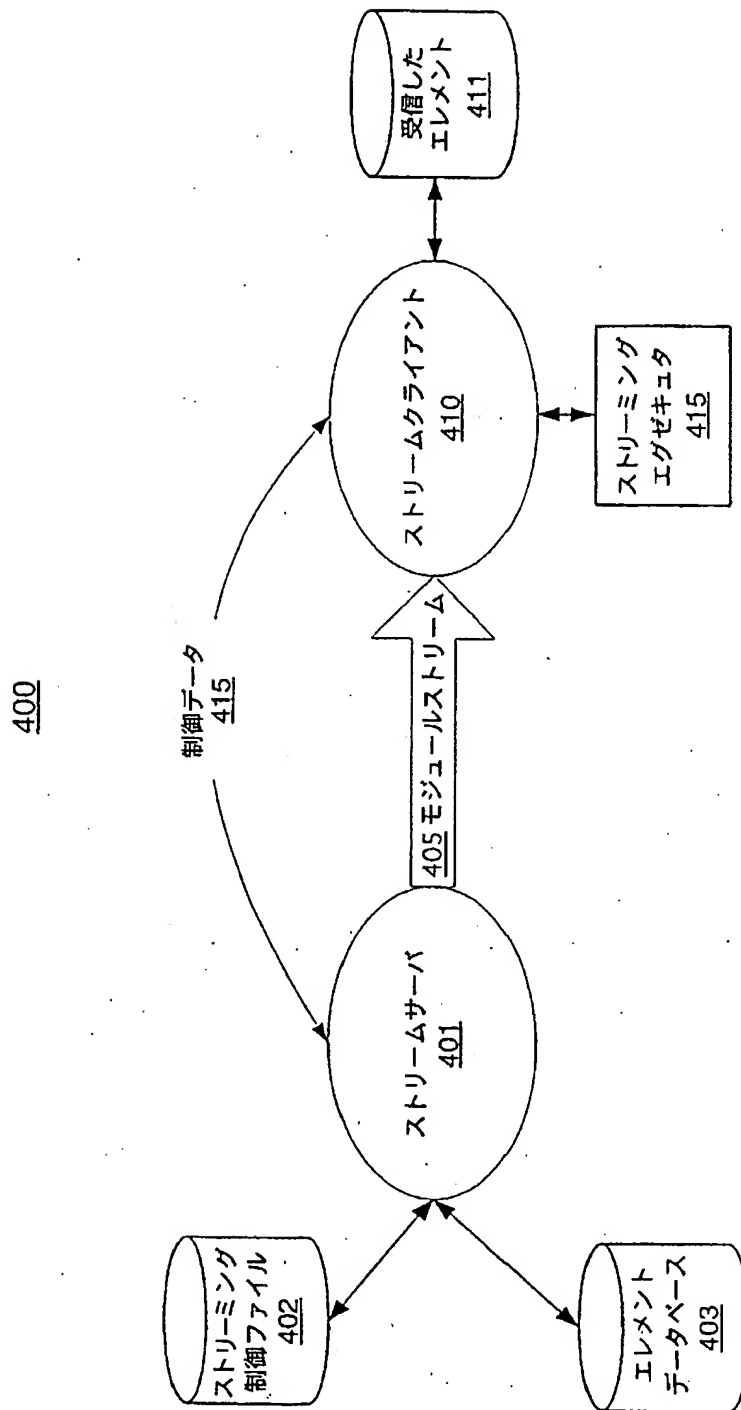
【图2】



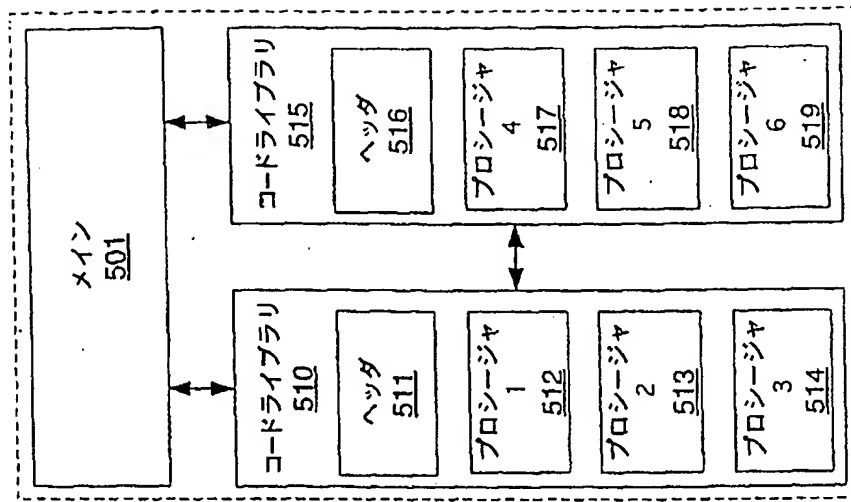
【図3】



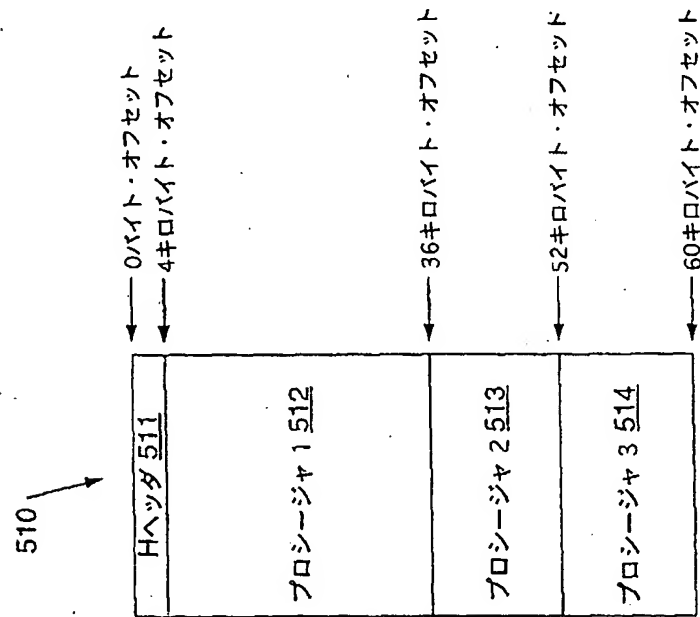
【図4】



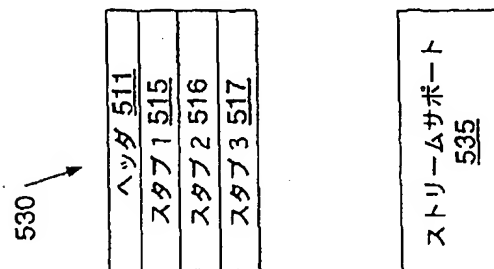
【図5A】



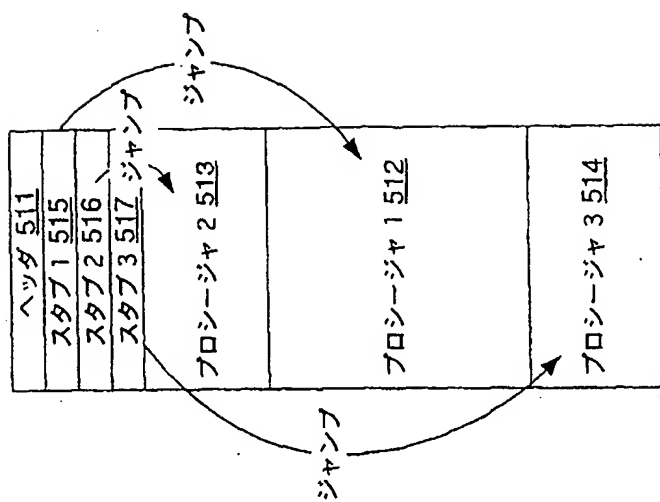
【図5B】



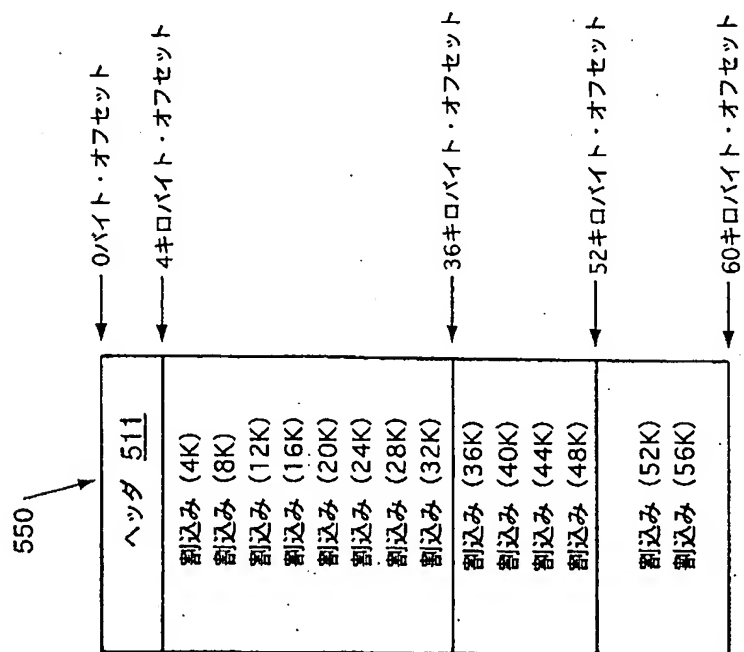
【図5C】



【図5D】



【図5E】



【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No.  
PCT/US99/16055

<b>A. CLASSIFICATION OF SUBJECT MATTER</b> IPC(7) : G06F 9/46 US CL : 709/310, 319 According to International Patent Classification (IPC) or to both national classification and IPC		
<b>B. FIELDS SEARCHED</b> Minimum documentation searched (classification system followed by classification symbols) U.S. : 709/310, 319 Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched Electronic data base consulted during the international search (name of data base and, where practicable, search terms used) EAST (BRS), IEEE (iconline) search terms: data streaming, module, class, Java, applet, www, predicted, future, expected, download		
<b>C. DOCUMENTS CONSIDERED TO BE RELEVANT</b>		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5,487,167 A (DINNALE et al.) 23 January 1996, cols 1,2,5-7	23
Y	US 5,742,768 A (GENNARO et al.) 21 April 1998, col 1 lines 38-63; cols 3-4	1-20, 24
Y	US 5,581,764 A (FITZGERALD et al.) 03 December 1996, col 10 lines 20-60, col 13 lines 20-38	21, 22
Y	GLASS, G. "A Universal Streaming Service" C++ Report April 1996 pp. 74-83	21, 22
Y	RITCHEY, T. "Java!" New Riders Publishing 1995 pp. 214-216	15
<input checked="" type="checkbox"/> Further documents are listed in the continuation of Box C. <input type="checkbox"/> See patent family annex.		
* Special categories of cited documents: "A" document defining the general state of the art which is not considered to be of particular relevance "B" earlier document published on or after the international filing date "L" document which may throw doubt on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "T" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "X" later document published after the international filing date in priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "Y" document of particular relevance, the claimed invention cannot be considered novel or obvious for reasons of its disclosure as an entire step when the document is taken alone "V" document of particular relevance, the claimed invention cannot be considered to involve an inventive step when the document is considered with one or more other such documents, such combination being obvious to a person skilled in the art "R" document member of the same patent family		
Date of the actual completion of the international search 13 APRIL 2000		Date of mailing of the international search report 16 MAY 2000
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230		Authorized officer Gary Emerson <i>James R. Matthews</i> Telephone No. (703) 305-9600

Form PCT/ISA/210 (second sheet) (July 1998)\*



## INTERNATIONAL SEARCH REPORT

International application No.  
PCT/US99/16053

## C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indicating, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5,765,164 A (PRASAD et al.) 09 June 1998, col 1 lines 20-25, col 4 lines 56 et seq., cols 7-9	1-20, 24
X	US 6,003,087 A (HOUSE, III et al.) 14 December 1999, col 3 line 41 through col 4 line 9	1-24

Form PCT/ISA/210 (continuation of second sheet) (July 1998)\*

フロントページの続き

(81)指定国 EP(AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OA(BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG), AP(GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), EA(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZA, ZW

(71)出願人 シュミュエル メラメード  
イスラエル ラマットーガン ネイヴェ  
ヤフシュア 35

(72)発明者 ユリ ラズ  
アメリカ合衆国 07410 ニュージャージー  
州 フェアローン ヒルサイド テラス  
36-02

(72)発明者 ヤフーダ フォルク  
イスラエル テルーアビブ レンブラント  
ストリート 10

(72)発明者 シュミュエル メラメード  
イスラエル ラマットーガン ネイヴェ  
ヤフシュア 35

Fターム(参考) 5B076 BB06

【要約の続き】

よび受信されたシーケンス内の第1モジュールをアプリケーションに統合するための手段を含む。第2コンピュータは、アプリケーションと関連付けられたモジュールのコレクションをストアするための手段、このコレクションからモジュールのシーケンスを選択するための手段、および第1コンピュータから第2コンピュータに選択されたシーケンスを転送するための手段を含む。